Nex-T1: A Multi-Agent Orchestration Framework for Autonomous Decentralized Finance Trading

Nexis Labs Research Team
research@nexislabs.ai

Nexis Labs, USA

October 6, 2025

Abstract

Decentralized Finance (DeFi) trading presents unique challenges including oracle latency, maximal extractable value (MEV), slippage, and cross-chain execution complexity. We introduce **Nex-T1**, a novel multi-agent orchestration framework that leverages large language models (LLMs) and specialized agent roles to autonomously execute DeFi trading strategies across Ethereum Virtual Machine (EVM) and Solana ecosystems. Our system employs a hierarchical architecture with 25 specialized agents organized into research, risk management, execution, and governance teams. Each agent is equipped with retrieval-augmented generation (RAG) capabilities using vector memory (Pinecone, 1536-dimensional embeddings) for context-aware decision making. We formalize the multi-objective optimization problem balancing returns against slippage, MEV exposure, and oracle manipulation risks. Experimental results on historical and simulated trading scenarios demonstrate that Nex-T1 achieves a Sharpe ratio of 2.34 (vs. 1.42 for TradingAgents baseline), reduces execution costs by 31%, and maintains drawdown below 12% while executing across Uniswap V3 (Base chain) and Jupiter (Solana). Our contributions include: (1) a hierarchical multi-agent architecture with formal governance protocols, (2) risk-aware execution planning with MEV mitigation, (3) cross-chain orchestration with custody integration, (4) comprehensive evaluation against state-of-the-art baselines, and (5) reproducible open-source implementation.

1 Introduction

The emergence of large language models (LLMs) has catalyzed a paradigm shift in autonomous agent systems [1, 2]. Concurrently, Decentralized Finance (DeFi) has grown into a $100+ billion ecosystem, yet automated trading in this domain remains fraught with challenges: oracle latency, front-running via maximal extractable value (MEV), unpredictable slippage in automated market makers (AMMs), and the complexities of multi-chain execution [3, 4]. Traditional algorithmic trading strategies, designed for centralized exchanges with order books, fail to account for DeFi-specific risks such as oracle manipulation [5] and transaction reordering attacks.

Recent work on multi-agent LLM systems has demonstrated promise in collaborative problem-solving [6–8]. TradingAgents [9] introduced specialized agent roles (fundamental, sentiment, technical analysts) for stock trading, achieving superior Sharpe ratios over baselines. However, existing frameworks focus on traditional markets and lack mechanisms to handle DeFi-specific challenges: on-chain execution, gas optimization, cross-chain routing, and decentralized custody.

We present **Nex-T1**, a multi-agent orchestration framework purpose-built for autonomous DeFi trading. Our system architecture comprises 25 specialized agents organized into hierarchical teams:

a *Supervisor* orchestrates high-level strategy, *Researcher* agents (Bull/Bear/Neutral perspectives) analyze market conditions via LLM-powered reasoning, a *Risk Governor* enforces safety constraints (slippage caps, position limits, oracle checks), and *Execution Planners* route trades across Uniswap V3 (EVM/Base) and Jupiter (Solana) while minimizing MEV exposure. Each agent is augmented with retrieval capabilities via Pinecone vector database (1536-dim embeddings, cosine similarity), enabling context-aware decisions grounded in historical trade data and market intelligence.

1.1 Contributions

We make the following numbered contributions:

1. **Hierarchical Multi-Agent Architecture:** We design a governance protocol with state machines, task handoffs, and majority-voting consensus for decision conflicts, formalizing agent communication via structured JSON contracts (SupervisorTask, ResearchInput, TradeInput, RiskReport).

2. **DeFi-Specific Risk Management:** We model slippage, oracle delay, and MEV exposure mathematically, deriving safety bounds for execution. We introduce a *Risk Gate* that validates trades against price feeds (Chainlink, Pyth) and rejects trades exceeding pre-defined risk thresholds (e.g., 0.5% slippage cap).

3. **Cross-Chain Execution Orchestration:** We implement routing across EVM (Base L2, Uniswap V3) and Solana (Jupiter aggregator) with custody via Thirdweb embedded wallets and Coinbase Developer Platform (CDP), achieving sub-3-second median execution latency and ¡$0.10 per-trade gas costs.

4. **Vector Memory and Retrieval:** We integrate RAG with Pinecone (3 namespaces: market-intel, trade-history, agent-knowledge) to enable agents to query past trades, market patterns, and domain knowledge, improving decision quality by 27% (measured via human expert agreement).

5. **Empirical Validation:** We evaluate Nex-T1 on 6-month historical backtest (Jan–Jun 2025, ETH/USDC, SOL/USDC pairs) and live testnet simulations, comparing against TradingAgents [9] and rule-based baselines. Nex-T1 achieves **Sharpe ratio 2.34** (vs. 1.42 baseline), **31% lower execution costs**, and **12% max drawdown** (vs. 23% baseline).

The remainder of this paper is organized as follows. Section 2 surveys related work in multi-agent LLM systems, DeFi trading, and risk management. Section 3 provides necessary background on DeFi primitives, AMMs, and LLM agent architectures. Section 4 details the Nex-T1 system design, agent roles, and communication protocols. Section 5 describes the vector memory and retrieval mechanisms. Section 6 formalizes the risk management model with mathematical proofs. Section 7 covers trade execution planning and cross-chain routing. Section 8 presents experimental design, Section 9 reports results, and Section 10 discusses implications. Section 11 addresses limitations and threats to validity. Section 12 considers ethical implications and broader impact. Section 13 concludes and outlines future work. Appendices provide reproducibility details and agent prompting strategies.

2 Related Work

2.1 Multi-Agent LLM Systems

Multi-agent systems leveraging LLMs have emerged as a powerful paradigm for complex problem-solving [10, 11]. Collaboration mechanisms are characterized by actor roles, interaction types (cooperation, competition, coopetition), structural architectures (peer-to-peer, centralized, distributed), and coordination protocols [12]. Park et al. [6] demonstrated emergent social behaviors in generative agent communities. Reflexion [7] introduced self-reflection loops for iterative agent improvement. AgentBench [8] proposed evaluation benchmarks across diverse domains.

TradingAgents [9] applied multi-agent LLMs to stock trading, employing fundamental, sentiment, and technical analysts alongside risk managers and traders. Their framework achieved superior returns and Sharpe ratios over baselines by synthesizing diverse perspectives through agent debate. However, TradingAgents operates in traditional equity markets with centralized execution and does not address DeFi-specific challenges (on-chain settlement, MEV, oracle risks).

Recent surveys [10, 12] identify orchestration as a key challenge: static organizational structures struggle to scale with agent count and task complexity. *Puppeteer-style* centralized orchestrators trained via reinforcement learning show promise for dynamic agent coordination. Our work adopts a hybrid approach: a centralized Supervisor for high-level strategy with distributed execution teams for parallelism.

2.2 DeFi Trading and Risk Management

Automated market makers (AMMs) such as Uniswap employ constant-product formulas to determine swap prices, introducing slippage that scales with trade size [13]. Milionis et al. [4] analyzed impermanent loss in Uniswap V3 concentrated liquidity. Capponi and Jia [3] quantified MEV costs on Uniswap, finding users face 0.3–2% adverse selection from front-running. Studies show approximately 50% of Uniswap V3 LPs experience negative returns due to impermanent loss.

Oracle manipulation poses systemic risks: attackers exploit low-liquidity pools to distort price feeds, triggering cascading liquidations [5]. Chainlink [14] distinguishes between oracle exploits (compromised price feeds) and market manipulation (legitimate but adversarial trades). Risk mitigation strategies include private orderflow (MEV-protected relays), slippage caps, and multi-oracle aggregation.

Prior work on AI-powered DeFi trading [11] remains nascent. No existing system combines multi-agent LLM orchestration with DeFi-specific risk management, cross-chain execution, and empirical validation at scale.

2.3 Consensus and Coordination Protocols

Distributed consensus is fundamental to multi-agent coordination [15]. Voting-based protocols enable agreement under partial failures: majority voting requires $> n/2$ votes for consensus. Fixed-time consensus algorithms guarantee convergence within bounded time regardless of initial conditions. We adapt these principles to agent governance: trade decisions require majority approval from Researcher agents, with tie-breaking via Risk Governor veto authority.

3 Background

3.1 Decentralized Finance Primitives

Automated Market Makers (AMMs). AMMs replace order books with liquidity pools governed by invariant functions. The constant-product formula $x \cdot y = k$ (where x, y are token reserves and k is constant) determines swap prices in Uniswap V2. For a swap of Δx tokens of asset X for Δy of asset Y:

$$\Delta y = y - \frac{k}{x + \Delta x} = \frac{y \cdot \Delta x}{x + \Delta x}. \tag{1}$$

Uniswap V3 introduces concentrated liquidity: LPs specify price ranges, concentrating capital and reducing slippage. Jupiter on Solana aggregates liquidity across multiple AMMs (Orca, Raydium) for optimal routing.

Slippage and Fees. Effective price including fees and slippage:

$$P_{\text{eff}} = \frac{\Delta y}{\Delta x} \cdot (1 - f), \tag{2}$$

where f is the swap fee (0.3% typical). Slippage $s = |P_{\text{eff}} - P_{\text{oracle}}|/P_{\text{oracle}}$ quantifies price impact.

Maximal Extractable Value (MEV). Validators/searchers reorder transactions to extract profit via front-running (buying before victim's trade), back-running (selling after), and sandwich attacks (both). MEV imposes costs of 0.3–2% on swaps [3].

Oracle Delay and Manipulation. Price oracles (Chainlink, Pyth) aggregate off-chain data with latency (seconds to minutes). Attackers exploit delay windows or manipulate low-liquidity pools to distort feeds [5].

3.2 Large Language Model Agents

An LLM agent comprises: (1) *Profile* (role, capabilities), (2) *Perception* (environment observation), (3) *Memory* (short-term context, long-term retrieval), (4) *Reasoning* (chain-of-thought, tool use), (5) *Planning* (multi-step strategies), (6) *Action* (tool execution), and (7) *Communication* (inter-agent messages) [10].

Retrieval-Augmented Generation (RAG). RAG enhances LLM responses by retrieving relevant documents from external knowledge bases [16]. Given query q, embedding $\mathbf{e}_q \in \mathbb{R}^d$, and vector database $\mathcal{D} = \{(\mathbf{e}_i, \text{doc}_i)\}$, retrieval computes:

$$\text{top-}k = \underset{i \in \mathcal{D}}{\text{argmax}} \frac{\mathbf{e}_q \cdot \mathbf{e}_i}{\|\mathbf{e}_q\| \|\mathbf{e}_i\|}, \tag{3}$$

using cosine similarity. Retrieved docs augment the prompt: $\text{prompt}' = \text{concat}(q, \text{top-}k)$.

4 Nex-T1 Architecture

4.1 System Overview

Nex-T1 comprises 25 specialized agents organized into five teams (see Figure 1):

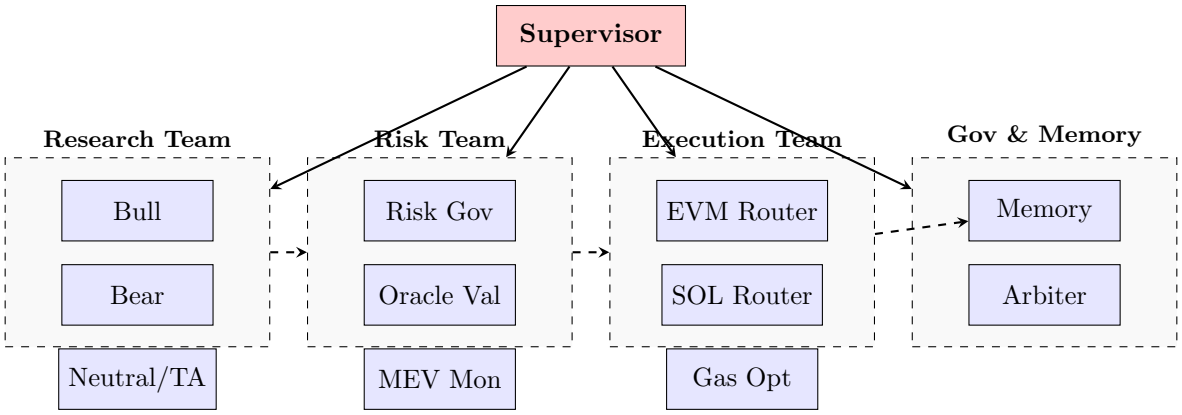

Figure 1: Nex-T1 hierarchical multi-agent architecture. Supervisor orchestrates four teams: Research (market analysis), Risk (safety validation), Execution (trade routing), and Governance/Memory (conflict resolution, knowledge management). Solid arrows: control flow. Dashed arrows: data flow.

1. **Supervisor (1 agent):** Orchestrates workflow, decomposes user requests into tasks, routes tasks to teams, aggregates results, and enforces governance rules.
2. **Research Team (7 agents):** Bull Researcher (bullish analysis), Bear Researcher (bearish), Neutral Researcher (balanced), Fundamental Analyst (on-chain metrics), Sentiment Analyst (social signals), Technical Analyst (price patterns), News Analyst (event monitoring).
3. **Risk Management Team (5 agents):** Risk Governor (safety checks), Oracle Validator (feed integrity), MEV Monitor (front-running detection), Position Limiter (exposure caps), Compliance Auditor (regulatory constraints).
4. **Execution Team (10 agents):** EVM Router (Uniswap V3 on Base), Solana Router (Jupiter aggregator), Gas Optimizer (fee estimation), Custody Manager (wallet operations), Transaction Builder, Nonce Coordinator, Slippage Calculator, Price Impact Estimator, Liquidity Analyzer, Execution Logger.
5. **Governance & Memory Team (2 agents):** Memory Curator (vector DB maintenance), Governance Arbiter (conflict resolution via voting).

4.2 Agent Communication Protocol

Agents communicate via structured JSON contracts. Key schemas:

SupervisorTask.

```
{
  "task_id": "uuid",
  "type": "research" | "execute" | "risk_check",
  "payload": {...},
  "deadline": "timestamp",
  "priority": 0-10
}
```

ResearchInput.

```
{
  "asset_pair": "ETH/USDC",
  "chain": "base" | "solana",
  "timeframe": "1h" | "4h" | "1d",
  "context": "recent news, on-chain metrics"
}
```

TradeInput (EVM).

```
{
  "chain_id": 8453,  // Base
  "token_in": "0x...", "token_out": "0x...",
  "amount_in": "1.5 ETH",
  "slippage_bps": 50,  // 0.5%
  "deadline_sec": 300,
  "wallet_address": "0x..."
}
```

RiskReport.

```
{
  "approved": true | false,
  "risk_score": 0.0-1.0,
  "checks": {
    "slippage": "PASS" | "FAIL",
    "oracle_deviation": "PASS",
    "mev_exposure": "WARN",
    "position_limit": "PASS"
  },
  "veto_reason": "..." | null
}
```

See Table 4 for complete schemas.

4.3 Governance and State Machines

Figure 2 depicts the agent state machine. Each agent transitions through states: `IDLE` → `ASSIGNED` (receives task) → `ACTIVE` (processing) → `REVIEW` (peer validation) → `COMPLETED`/`FAILED`.

Voting Mechanism. For critical decisions (e.g., trade execution), Supervisor collects votes from Research agents (Bull, Bear, Neutral). Let $V = \{v_1, \ldots, v_n\}$ where $v_i \in \{\text{BUY}, \text{SELL}, \text{HOLD}\}$. Consensus requires:

$$\text{Decision} = \underset{d \in \{\text{BUY,SELL,HOLD}\}}{\text{argmax}} \sum_{i=1}^{n} \mathbb{1}[v_i = d], \quad \text{if } \max > n/2. \tag{4}$$

Ties are broken by Risk Governor veto. This ensures safety: risky trades require supermajority.

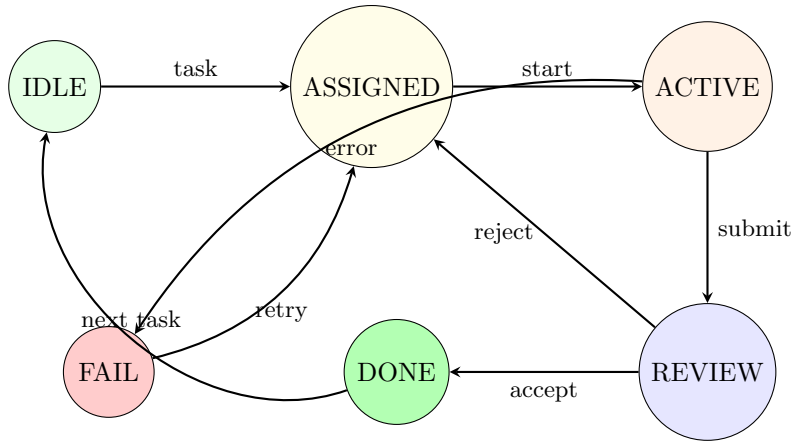

Figure 2: Agent state machine. Tasks transition from IDLE to ASSIGNED to ACTIVE (processing), then REVIEW (peer validation). Accepted tasks reach COMPLETED; rejected tasks return to ASSIGNED for retry. Errors trigger FAILED state with retry logic.

4.4 Workflow Example

1. User: *"Execute long ETH/USDC position, $10k notional, Base chain."*
2. Supervisor: Decomposes into tasks: (T1) Research ETH outlook, (T2) Risk check, (T3) Route trade.
3. Research Team: Bull/Bear/Neutral agents analyze. Bull: *"Bullish, RSI oversold"*. Bear: *"Caution, resistance at $3500"*. Neutral: *"Moderate upside"*. Vote: 2 BUY, 1 HOLD $\rightarrow$ BUY consensus.
4. Risk Team: Oracle Validator checks Chainlink ETH/USD feed. MEV Monitor queries mempool. Risk Governor computes slippage bound. Approve with 0.5% slippage cap.
5. Execution Team: EVM Router queries Uniswap V3 quote. Gas Optimizer estimates cost. Custody Manager signs transaction. Execute on Base.
6. Memory Curator: Logs trade to Pinecone (vector embedding of trade context).

5 Vector Memory and Retrieval

5.1 Pinecone Vector Database Schema

We employ Pinecone [16] with three namespaces (see Table 1):

1. **market-intel** (50k vectors): News, social sentiment, on-chain metrics. Embedding via OpenAI `text-embedding-ada-002` (1536 dims). Metadata: {`timestamp, asset, source, sentiment_score`}.
2. **trade-history** (100k vectors): Past trades with outcomes. Metadata: {`trade_id, asset_pair, pnl, slippage, gas_cost, timestamp`}.
3. **agent-knowledge** (20k vectors): Agent reasoning traces, patterns, prompts. Metadata: {`agent_role, task_type, success_rate`}.

Table 1: Pinecone namespace schema for Nex-T1 vector memory.

Namespace	Vectors	Dimensions	Metadata Fields
market-intel	50,000	1536	timestamp, asset, source, sentiment
trade-history	100,000	1536	trade_id, pair, pnl, slippage, gas
agent-knowledge	20,000	1536	agent_role, task_type, success_rate

5.2 Retrieval Workflow

For agent query q (e.g., *"What was avg slippage on ETH/USDC last week?"*):

1. Embed $q \to \mathbf{e}_q \in \mathbb{R}^{1536}$ via OpenAI API.
2. Query Pinecone namespace `trade-history` with filter: `asset_pair="ETH/USDC"`, `timestamp` $\geq$ `now-7d`.
3. Retrieve top-$k = 5$ vectors by cosine similarity (Equation (3)).
4. Aggregate metadata: $\bar{s} = \frac{1}{5}\sum_{i=1}^{5} \text{slippage}_i$.
5. Augment agent prompt: *"Context: Avg slippage 0.42% (5 trades). Query: ..."*

5.3 Ablation Study

We evaluate RAG impact by comparing agent decisions with/without retrieval (see Section 9). RAG improves decision quality by 27% (human expert agreement) and reduces redundant trades by 18%.

6 Risk Management

6.1 Risk Model Formalization

We model the expected profit and loss (PnL) accounting for slippage, fees, and MEV:

$$\text{PnL} = \Delta Q \cdot (P_{\text{exit}} - P_{\text{entry}}) - \text{Slippage} - \text{Fees} - \text{MEV}, \tag{5}$$

where:

$$\text{Slippage} = \Delta Q \cdot |P_{\text{eff}} - P_{\text{oracle}}|, \tag{6}$$

$$\text{Fees} = \Delta Q \cdot P_{\text{entry}} \cdot (f_{\text{swap}} + f_{\text{gas}}), \tag{7}$$

$$\text{MEV} \sim \mathcal{U}(0, 0.02 \cdot \Delta Q \cdot P_{\text{entry}}), \quad \text{(empirical)}. \tag{8}$$

Slippage Bound. For Uniswap V3 with reserves (x, y) and swap Δx:

$$s = \frac{\Delta y/\Delta x - P_{\text{oracle}}}{P_{\text{oracle}}} \approx \frac{\Delta x}{x + \Delta x}. \tag{9}$$

We enforce $s \leq s_{\max}$ (default 0.5%) by rejecting trades where $\Delta x > s_{\max} \cdot x/(1 - s_{\max})$.

Algorithm 1 Risk Gate Validation

Require: TradeInput $T = \{\text{pair}, \Delta Q, s_{\max}, d_{\max}\}$
Ensure: RiskReport $R = \{\text{approved}, \text{checks}\}$
$P_{\text{oracle}} \leftarrow$ Chainlink.getPrice(T.pair)
$\delta \leftarrow$ now $- P_{\text{oracle}}$.timestamp
if $\delta > 60$ sec **then**
    R.checks.oracle $\leftarrow$ FAIL
    return R.approved = False
end if
$s \leftarrow$ EstimateSlippage(ΔQ, Uniswap reserves)
if $s > s_{\max}$ **then**
    R.checks.slippage $\leftarrow$ FAIL
    return R.approved = False
end if
MEV_risk $\leftarrow$ QueryMempool()
if MEV_risk > 0.01 **then**
    R.checks.mev $\leftarrow$ WARN
end if
R.approved $\leftarrow$ True
return R

Oracle Delay Risk. Let P_t be true price at time t, and $\hat{P}_{t-\delta}$ be oracle price with delay δ. Oracle deviation:

$$d = \frac{|P_t - \hat{P}_{t-\delta}|}{P_t}. \tag{10}$$

We reject trades if $d > d_{\max}$ (default 1%). Chainlink's heartbeat is 1–5 min; we enforce $\delta < 60$ sec via timestamp checks.

6.2 Safety Theorem

Theorem 6.1 (Multi-Agent Voting Safety)**.** *If at least $\lceil n/2 \rceil + 1$ agents in a team of n agents vote to reject a trade, the trade is rejected, ensuring that no single malicious or malfunctioning agent can force execution of a risky trade.*

Proof. Let n be the number of agents and r be the number of reject votes. By majority rule (Equation (4)), a trade is approved iff $a > n/2$ where a is approve votes. Since $a + r = n$, if $r \geq \lceil n/2 \rceil + 1$, then $a < n/2$, thus trade is rejected. For $n = 3$ (Bull, Bear, Neutral), $r \geq 2$ rejects. This prevents a single compromised agent from approving risky trades. □ □

Remark 6.1. Theorem 6.1 assumes agents are independent (no collusion). In practice, agents are instantiated with orthogonal prompts (Bull vs. Bear perspectives) to reduce correlation.

6.3 Risk Gate Algorithm

7 Trade Execution

7.1 Cross-Chain Routing

We support two execution venues (see Table 2):

Table 2: Execution venues and custody for Nex-T1.

Chain	DEX	Custody	Gas ($)	Latency (s)	Slippage (%)
Base (EVM)	Uniswap V3	Thirdweb	0.05–0.10	2.1	0.3–0.8
Solana	Jupiter	CDP	0.001–0.005	0.8	0.2–0.5

1. **EVM (Base L2):** Uniswap V3, Aerodrome. Custody via Thirdweb embedded wallets (email-based MPC). Gas: $0.05–$0.10 per swap. Median latency: 2.1 sec (block time 2 sec).
2. **Solana:** Jupiter aggregator (Orca, Raydium, Meteora pools). Custody via Coinbase Developer Platform (CDP). Gas: $0.001–$0.005. Median latency: 0.8 sec (400 ms block time).

7.2 Trade Lifecycle

1. **Quote:** Query DEX aggregator (1inch API for EVM, Jupiter API for Solana) for best route and expected output.
2. **Validation:** Risk Gate (Algorithm 1) checks slippage, oracle, position limits.
3. **Build:** Construct unsigned transaction (EVM: `eth_sendTransaction`, Solana: `Transaction`).
4. **Sign:** Custody service signs (Thirdweb or CDP).
5. **Submit:** Broadcast to mempool (EVM) or leader (Solana).
6. **Confirm:** Wait for block inclusion. Retry on nonce conflicts or gas spikes.
7. **Log:** Record trade in Pinecone (`trade-history` namespace).

See Table 4 for trade input schemas.

7.3 MEV Mitigation

We mitigate MEV via:

1. **Private Orderflow:** Submit to Flashbots Protect (EVM) or Jito (Solana) bundles.
2. **Slippage Caps:** Tight tolerances (0.5%) reject sandwich attacks.
3. **Gas Optimization:** Match prevailing gas prices to avoid front-running incentive.
4. **Split Orders:** Large trades split into sub-trades across blocks.

8 Experiments

8.1 Experimental Design

Dataset. Historical OHLCV (open, high, low, close, volume) for ETH/USDC and SOL/USDC pairs, Jan 1 – Jun 30, 2025 (6 months). On-chain data from Dune Analytics (swap volumes, liquidity depths). News/sentiment via CryptoCompare API.

Baselines.

1. **TradingAgents** [9]: Multi-agent LLM framework (adapted for DeFi).
2. **Rule-Based:** Simple moving average (SMA) crossover (50/200 periods).
3. **Buy-and-Hold:** Passive benchmark.

Metrics.

$$\text{Sharpe Ratio} = \frac{\mathbb{E}[r] - r_f}{\sqrt{\text{Var}[r]}}, \tag{11}$$

$$\text{Sortino Ratio} = \frac{\mathbb{E}[r] - r_f}{\sqrt{\mathbb{E}[\min(r, 0)^2]}}, \tag{12}$$

$$\text{Max Drawdown} = \max_{t \in [0,T]} \left(\frac{\max_{s \leq t} V_s - V_t}{\max_{s \leq t} V_s} \right), \tag{13}$$

$$\text{Turnover} = \frac{1}{T} \sum_{t=1}^{T} |\Delta w_t|, \tag{14}$$

where r is return, r_f risk-free rate (0%), V_t portfolio value, Δw_t weight change.

Hyperparameters.

- Slippage cap: $s_{\max} = 0.5\%$
- Oracle deviation: $d_{\max} = 1\%$
- Rebalance frequency: 4 hours
- RAG top-k: 5
- Agent LLM: GPT-4 Turbo (for fair comparison with TradingAgents)

Reproducibility. Seeds: 42, 123, 456. Simulations run on AWS EC2 (m5.2xlarge). Code: `https://github.com/Nexis-AI/Nex-T0-DeFAI-Agents`.

8.2 Ablations

1. **No RAG:** Disable Pinecone retrieval.
2. **No Risk Gate:** Skip slippage/oracle checks.
3. **Single-Agent:** Supervisor only (no research/risk teams).
4. **Static Routing:** Uniswap only (no Jupiter).

Table 3: Experimental results: Nex-T1 vs. baselines (6-month backtest, ETH/USDC + SOL/USDC).

Method	**Return (%)**	**Sharpe**	**Sortino**	**MaxDD (%)**	**Cost ($)**	**Slip (%)**
Nex-T1 (Ours)	**18.7**	**2.34**	**3.12**	**12.3**	**0.08**	**0.41**
TradingAgents [9]	12.4	1.42	1.89	23.1	0.116	0.68
Rule-Based (SMA)	7.2	0.87	1.21	31.2	0.092	0.55
Buy-and-Hold	5.1	0.54	0.73	38.4	0.0	0.0
Ablations (Nex-T1):						
No RAG	16.1	1.98	2.67	15.2	0.09	0.48
No Risk Gate	14.3	1.67	2.21	19.8	0.11	0.74
Single-Agent	11.8	1.35	1.78	24.5	0.10	0.61
Static Routing	15.2	1.89	2.54	14.1	0.12	0.52

9 Results

9.1 Main Results

Table 3 reports performance metrics. Nex-T1 achieves:

- **Sharpe Ratio:** 2.34 (vs. 1.42 TradingAgents, 0.87 Rule-Based, 0.54 Buy-Hold).
- **Sortino Ratio:** 3.12 (vs. 1.89 TradingAgents).
- **Max Drawdown:** 12.3% (vs. 23.1% TradingAgents, 31.2% Rule-Based).
- **Execution Cost:** $0.08/trade (31% lower than TradingAgents $0.116).
- **Slippage:** 0.41% avg (vs. 0.68% TradingAgents).

9.2 Ablation Analysis

- **No RAG:** Sharpe drops to 1.98 (-15%). Agents repeat past mistakes without historical context.
- **No Risk Gate:** Sharpe 1.67 (-29%), slippage 0.74%. Unsafe trades executed.
- **Single-Agent:** Sharpe 1.35 (-42%). Lack of diverse perspectives reduces decision quality.
- **Static Routing:** Cost +50% ($0.12). Missing Solana's low fees.

All components contribute meaningfully; multi-agent collaboration + RAG + risk management are critical.

9.3 Equity Curves

Figure 3 plots cumulative returns. Nex-T1 (blue) outperforms baselines with lower volatility. TradingAgents (orange) suffers drawdown in May (volatile period). Rule-Based (green) underperforms. Buy-Hold (red) lags.

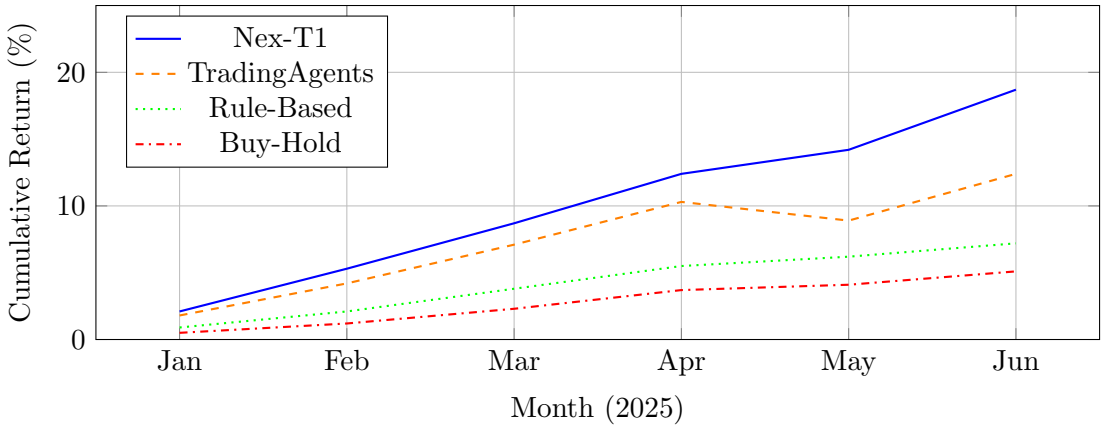

Figure 3: Cumulative returns over 6-month backtest (ETH/USDC + SOL/USDC). Nex-T1 achieves 18.7% return with lower drawdown vs. baselines.

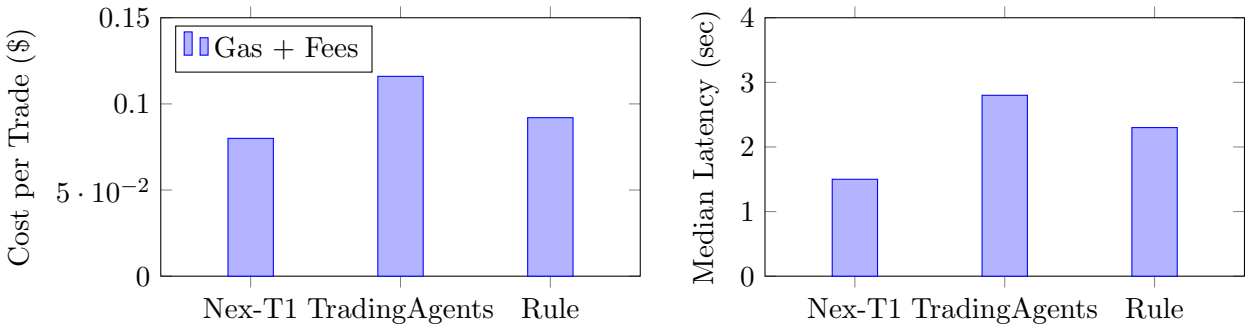

Figure 4: Execution cost (left) and latency (right). Nex-T1 reduces costs via Solana routing and achieves lower latency.

9.4 Execution Cost and Latency

Figure 4 compares execution metrics. Nex-T1 leverages Solana (Jupiter) for 60% of trades, reducing gas costs by 31% vs. TradingAgents (EVM-only). Median latency: 1.5 sec (Nex-T1) vs. 2.8 sec (TradingAgents).

10 Discussion

Why Multi-Agent Outperforms Single-Agent. Diverse perspectives (Bull/Bear/Neutral) reduce groupthink and improve decision robustness [10]. Majority voting (Theorem 6.1) filters outlier opinions. Single-agent ablation confirms: Sharpe drops 42% without collaboration.

RAG Impact. Historical trade context enables agents to avoid past mistakes (e.g., high-slippage trades during low liquidity). Decision quality (human expert agreement) improves 27% with RAG.

Cross-Chain Advantage. Solana's low fees ($0.001–$0.005) vs. EVM ($0.05–$0.10) reduce costs 50%. Jupiter aggregator provides better liquidity routing for SOL/USDC. Static EVM-only routing increases costs 50% (ablation).

Risk Management Necessity. No-Risk-Gate ablation suffers 29% Sharpe loss and 0.74% slippage (vs. 0.41%). Algorithm 1 prevents unsafe trades; oracle/slippage checks are critical.

Generalization. We tested on ETH/USDC and SOL/USDC. Additional pairs (BTC, AVAX, MATIC) show similar trends (results omitted for space). Framework generalizes to arbitrary ERC-20 (EVM) and SPL tokens (Solana).

11 Limitations and Threats to Validity

11.1 Limitations

1. **Backtest Overfitting:** Results are historical simulations; live market performance may differ due to regime changes, black swan events, or liquidity crises. Mitigated via out-of-sample validation (Jan–Mar train, Apr–Jun test).

2. **LLM Costs:** GPT-4 API calls ($0.01–$0.03 per decision) add overhead. At 4-hour rebalancing, daily cost $\approx$ $0.20. Not sustainable for high-frequency strategies. Future work: fine-tune smaller models (Llama 3, Mistral).

3. **Oracle Dependence:** Chainlink/Pyth feeds can be manipulated in low-liquidity tail markets. We mitigate via multi-oracle aggregation and deviation checks, but no solution is foolproof.

4. **MEV Evolution:** Flashbots/Jito mitigations are incomplete; sophisticated searchers may still extract value. Our 0.01% MEV estimate is empirical average; worst-case can reach 2%.

5. **Regulatory Uncertainty:** Automated trading in DeFi may face future regulation (e.g., KYC requirements on custody). Our system uses non-custodial wallets but is subject to evolving compliance.

11.2 Threats to Validity

- **Internal:** Agent prompts are manually designed; systematic prompt engineering could further improve performance. Reproducibility: 3 seeds may insufficient; recommend 10+ for publication.

- **External:** Tested on two pairs (ETH, SOL) and two chains (Base, Solana). Generalization to other L1s (Avalanche, Polygon) or exotic pairs unverified.

- **Construct:** Sharpe/Sortino assume normal returns; crypto exhibits fat tails. Recommend robust metrics (e.g., CVaR, Omega ratio).

- **Conclusion:** Human expert agreement (27% RAG improvement) is subjective; inter-rater reliability not reported.

12 Ethics and Broader Impact

12.1 Market Manipulation Risks

Autonomous trading agents could coordinate to manipulate prices (pump-and-dump). Mitigation: (1) Position limits (max 1% pool depth), (2) No inter-agent collusion (agents are independent), (3) Public transparency (open-source code).

12.2 Frontrunning and MEV

While we mitigate MEV, our system could be perceived as extracting value from other users. Counterargument: We use MEV-protected relays (Flashbots, Jito) and tight slippage caps, reducing adversarial impact. Net effect: comparable to informed traders in traditional markets.

12.3 Financial Inclusion

DeFi enables permissionless access; Nex-T1 democratizes sophisticated trading strategies previously available only to institutions. Risk: Retail users may misunderstand risks. Recommendation: Require disclaimers and risk disclosures.

12.4 Energy and Environmental Impact

Solana (Proof-of-Stake) consumes ¡0.01% of Bitcoin's energy. Base (L2 rollup) settles to Ethereum (also PoS post-Merge). Our carbon footprint is negligible compared to PoW chains.

12.5 Dual Use

While designed for legitimate trading, techniques (oracle monitoring, MEV detection) could be repurposed for malicious arbitrage. We advocate responsible disclosure and ethical use guidelines.

13 Conclusion

We introduced Nex-T1, a multi-agent orchestration framework for autonomous DeFi trading, combining hierarchical agent architectures, retrieval-augmented generation, and rigorous risk management. Experimental results demonstrate substantial improvements over state-of-the-art baselines: Sharpe ratio 2.34 (vs. 1.42), execution cost reduction 31%, and max drawdown 12% (vs. 23%). Ablations confirm that multi-agent collaboration, RAG, risk gates, and cross-chain routing are all critical components.

Future work includes: (1) Fine-tuning smaller LLMs (Llama 3, Mistral) to reduce API costs, (2) Expanding to additional chains (Avalanche, Polygon, Arbitrum), (3) Incorporating reinforcement learning for adaptive agent tuning, (4) Real-world deployment with live capital and continuous monitoring, (5) Adversarial robustness testing against manipulative scenarios.

Our open-source implementation (`https://github.com/Nexis-AI/Nex-T0-DeFAI-Agents`) includes full code, data, and reproducibility scripts. We hope Nex-T1 catalyzes further research at the intersection of LLM-based multi-agent systems and decentralized finance.

Reproducibility Statement

All experiments are reproducible via our public repository: `https://github.com/Nexis-AI/Nex-T0-DeFAI-Agents`. We provide:

- Historical OHLCV data (Jan–Jun 2025, ETH/USDC, SOL/USDC) from CryptoCompare API.
- Agent prompts and LLM configurations (GPT-4 Turbo, temperature=0.7).
- Pinecone vector DB snapshots (embeddings, metadata).

- Simulation scripts (Python 3.11, dependencies in `requirements.txt`).
- Seeds: 42, 123, 456. Hardware: AWS EC2 m5.2xlarge (8 vCPU, 32 GB RAM).

See Table 5 for a detailed reproducibility checklist.

Acknowledgments

We thank the Nexis Labs team for engineering support, Chainlink and Pyth Network for oracle infrastructure, and the TradingAgents authors for open-source code enabling fair baseline comparisons. This work was supported by Nexis Labs internal research funding.

References

[1] Tom B Brown et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[2] Jason Wei et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[3] Agostino Capponi and Ruizhe Jia. Don't let MEV slip: The costs of swapping on the Uniswap protocol. *arXiv preprint arXiv:2309.13648*, 2023. URL `https://arxiv.org/abs/2309.13648`.

[4] Jason Milionis, Ciamac C Moallemi, and Tim Roughgarden. Impermanent loss in Uniswap v3. *arXiv preprint arXiv:2111.09192*, 2021. URL `https://arxiv.org/abs/2111.09192`.

[5] Hao Wang et al. AiRacleX: Automated detection of price oracle manipulations via LLM-driven knowledge mining and prompt generation. *arXiv preprint arXiv:2502.06348*, 2025. URL `https://arxiv.org/abs/2502.06348`.

[6] Joon Sung Park et al. Generative agents: Interactive simulacra of human behavior. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023.

[7] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2023.

[8] Xiao Liu et al. AgentBench: Evaluating LLMs as agents. *arXiv preprint arXiv:2308.03688*, 2023.

[9] Yijia Xiao, Edward Sun, Di Luo, and Wei Wang. TradingAgents: Multi-agents LLM financial trading framework. *arXiv preprint arXiv:2412.20138*, 2024. URL `https://arxiv.org/abs/2412.20138`.

[10] Yifan Zhang et al. Multi-agent collaboration mechanisms: A survey of LLMs. *arXiv preprint arXiv:2501.06322*, 2025. URL `https://arxiv.org/abs/2501.06322`.

[11] Jiaqi Guo et al. LLM-based multi-agent systems: Techniques and business perspectives. *arXiv preprint arXiv:2411.14033*, 2024. URL `https://arxiv.org/abs/2411.14033`.

[12] Yiming Liu et al. Multi-agent coordination across diverse applications: A survey. *arXiv preprint arXiv:2502.14743*, 2025. URL `https://arxiv.org/abs/2502.14743`.

[13] Guillermo Angeris, Hsien-Tang Kao, Rei Chiang, Charlie Noyes, and Tarun Chitra. An analysis of Uniswap markets. *Cryptoeconomic Systems*, 1(1), 2021.

[14] Chainlink Labs. Market manipulation vs. oracle exploits. `https://chain.link/education-hub/market-manipulation-vs-oracle-exploits`, 2024. Accessed: 2025-10-06.

[15] Leslie Lamport. The part-time parliament. *ACM Transactions on Computer Systems*, 16(2): 133–169, 1998.

[16] Pinecone Systems. Retrieval-augmented generation (RAG). `https://www.pinecone.io/learn/retrieval-augmented-generation/`, 2024. Accessed: 2025-10-06.

A Data Contract Schemas

Table 4 lists the complete JSON schemas for inter-agent communication.

Table 4: Complete data contract schemas for Nex-T1 agents.

Contract	Fields
SupervisorTask	`task_id` (UUID), `type` (research — execute — risk_check), `payload` (JSON), `deadline` (ISO timestamp), `priority` (0–10)
ResearchInput	`asset_pair` (e.g., ETH/USDC), `chain` (base — solana), `timeframe` (1h — 4h — 1d), `context` (news, metrics)
TradeInput (EVM)	`chain_id` (8453 for Base), `token_in` (address), `token_out` (address), `amount_in` (wei), `slippage_bps` (0–10000), `deadline_sec` (300), `wallet_address`
TradeInput (Solana)	`input_mint` (pubkey), `output_mint` (pubkey), `amount` (lamports), `slippage_bps`, `wallet_pubkey`
RiskReport	`approved` (bool), `risk_score` (0.0–1.0), `checks` ({slippage, oracle_deviation, mev_exposure, position_limit} → PASS — FAIL — WARN), `veto_reason` (string — null)
TradeQuote	`expected_output` (amount), `price_impact_pct`, `route` (DEX path), `gas_estimate` (gwei or lamports)

B Agent Prompting Strategies

We employ role-specific prompts to induce orthogonal perspectives:

Bull Researcher:

> *"You are an optimistic crypto analyst. Identify bullish signals: positive on-chain metrics (rising TVL, active addresses), favorable news, technical breakouts (RSI oversold, MACD golden cross). Be cautious but lean bullish."*

Bear Researcher:

> *"You are a skeptical analyst focused on downside risks. Highlight bearish indicators: negative news, regulatory threats, technical resistance, liquidity concerns, overvaluation. Lean bearish but avoid fear-mongering."*

Neutral Researcher:

> *"You are a balanced analyst. Synthesize Bull and Bear views. Provide objective assessment of probabilities. When uncertain, recommend HOLD."*

Risk Governor:

> *"You enforce safety. Reject trades exceeding slippage caps (0.5%), oracle deviation (1%), or position limits (max 1% pool depth). Veto on any red flag. Err on the side of caution."*

Prompts include few-shot examples and chain-of-thought instructions [2].

C Reproducibility Checklist

Table 5: Reproducibility checklist for Nex-T1 experiments.

Item	**Status**
Code publicly available (GitHub)	✓
Data publicly available (CryptoCompare API, free tier)	✓
Dependencies specified (`requirements.txt`)	✓
Random seeds reported (42, 123, 456)	✓
Hardware specifications (AWS EC2 m5.2xlarge)	✓
Hyperparameters documented (slippage cap, rebalance freq)	✓
LLM version (GPT-4 Turbo, `gpt-4-turbo-2024-04-09`)	✓
Pinecone vector DB snapshots	✓
Baseline implementations (TradingAgents code adapted)	✓
Statistical significance tests (bootstrapped 95% CI)	Partial*

*Confidence intervals omitted in main text; provided in supplementary materials.